**Computer Graphics Function**

**Iitgraph**

The initgraph function is used to switch the output from text mode to graphics mode.

The initgraph function takes three arguments.

**Declaration**: initgraph(&dr , &md , ”c:\\tc\\bgi” );

**Cleardevice**

The cleardevice function clears the screen in graphics mode and sets the current position to (0,0). Clearing the screen consists of filling the screen with current background color.  
  
**Declaration**: void cleardevice();

**Example**: cleardevice();

**Line**

Line function is used to draw a line from a point(x1,y1) to point(x2,y2) i.e. (x1,y1) and (x2,y2) are end points of the line. The code given below draws a line.  
 **Declaration**: void line(int x1, int y1, int x2, int y2);  
**Example**: line(100,100,200,200)

**Circle**

Circle function is used to draw a circle with center (x,y) and third parameter specifies the radius of the circle. The code given below draws a circle.  
  
**Declaration**: void circle(int x, int y, int radius.  
**Example:** circle(200,200,100)

**Rectangle**

The rectangle function is used to draw a rectangle. Coordinates of left top and right bottom corner are required to draw the rectangle. left specifies the X-coordinate of top left corner, top specifies the Y-coordinate of top left corner, right specifies the X-coordinate of right bottom corner, bottom specifies the Y-coordinate of right bottom corner. The code given below draws a rectangle  
  
**Declaration**: void rectangle(int left, int top, int right, int bottom);  
**Example**: rectangle (200,200,300,300)

**Outtextxy**

The outtextxy function display text or string at a specified point(x,y) on the screen.  
  
**Declaration**: void outtextxy(int x, int y, char \*string);  
x, y are coordinates of the point and third argument contains the address of string to be displayed.

**Example**: outtextxy(100,100,”HELLO”)

**Getch()**

The getch() is nonstandard function and is present in conio.h header file.

Line these functions, getch() also reads a single character from keyboard. But it does not use any buffer, so the entered character is immediately returned without waiting for the enter key.

**Declaration**: int getch(void);

**Example**:

#include<stdio.h>

#include<conio.h>

Int main()

{

printf( , getch());

return 0;

}

**Input**: g (without enter key)

**Output**: Program terminates immediately. But when you use DOS shell in Turbo C, it shows a single g, i.e., ‘g’

**Closegraph()**

The closegraph function closes the graphics mode, deallocates all memory allocated by graphics system and restores the screen to the mode it was in before you called initgraph.  
  
**Declaration**: void closegraph()

**Example**: closegraph();

**Setcolor**

The header file graphics.h contains setcolor() function which is used to set the current drawing color to the new color.

**Syntax**: void setcolor(int color);

**COLOR INT VALUES**

BLACK 0

BLUE 1

GREEN 2

CYAN 3

RED 4

MAGENTA 5

BROWN 6

LIGHTGRAY 7

DARKGRAY 8

LIGHTBLUE 9

LIGHTGREEN 10

LIGHTCYAN 11

LIGHTRED 12

LIGHTMAGENTA 13

YELLOW 14

WHITE 15

**Ellipse**

Ellipse is used to draw an ellipse (x,y) are coordinates of center of the ellipse, stangle is the starting angle, end angle is the ending angle, and fifth and sixth parameters specifies the X and Y radius of the ellipse. To draw a complete ellipse strangles and end angle should be 0 and 360 respectively.  
  
**Declarations**: void ellipse(int x, int y, int stangle, int endangle, int xradius, int yradius).  
**Example**: ellipse (100,100,0,360,50,25);

**Setbkcolor**

Setbkcolor function changes current background color.

**Declaration**: void setbkcolor(int color);

**Example**: setbkcolor(YELLOW) changes the current background color to YELLOW.

**Arc**

The "arc" function is used to draw an arc with center (x, y) and stangle specifies starting angle, endangle specifies the end angle and last parameter specifies the radius of the arc. arc function can also be used to draw a circle but for that starting angle and end angle should be 0 and 360 respectively  
  
**Declaration**: void arc(int x, int y, int stangle, int endangle, int radius);  
**Example**: arc100,100,0,135,50)

**Setfillstyle**

The header file graphics.h contains setfillstyle() function which sets the current fill pattern and fill color.  
  
**Declaration**: void setfillstyle( int pattern, int color);

**Example**: pattern = HACTH\_FILL, Color = RED

Circle : x=250, y=250, radius=100

**Floodfill**

The floodfill function is used to fill an enclosed area. Current fill pattern and fill color is used to fill the area.(x, y) is any point on the screen if (x,y) lies inside the area then inside will be filled otherwise outside will be filled,border specifies the color of boundary of area. To change fill pattern and fill color use setfillstyle. Code given below draws a circle and then fills  
  
**Declaration**: void floodfill(int x, int y, int border);  
**Example**: floodfill(100,100,GREEN);

**Cout**

The cout object in C++ is on ostreams, it is used to display the output to the standard output device.

**Declaration**: cout << varName;

**Getmaxcolor**

The `getmaxcolor function returns maximum color value for current graphics mode and driver. Total number of colors available for current graphics mode and driver are ( getmaxcolor() + 1 ) as color numbering starts from zero.  
  
**Declaration**: int getmaxcolor();

**Getmaxx**

The getmaxx function returns the maximum X coordinate for current graphics mode and driver.  
  
**Declaration**: int getmaxx()

**Example**: maxx=getmaxx();

**Getmaxy**

The getmaxy function returns the maximum Y coordinate for current graphics mode and driver.  
  
**Declaration**: int getmaxy();

**Example**: maxy=getmaxy();

**Return0**

The main function is generally supposed to return a value and after it returns something it finishes execution. The return 0 mean success and returning a non zero number means failure. Thus we “return 0” at the end to main function. But you can run the main function without the return 0. It works the same.

**Declaration**: return 0;

**Putpixel**

The putpixel function plots a pixel at location (x, y) of specified color.  
  
**Declaration**: void putpixel(int x, int y, int color);  
**Example**:putpixel(100,100,BLUE)

If we want to draw a GREEN color pixel at (35, 45) then we will write putpixel(35, 35, GREEN); in our c program, putpixel function can be used to draw circles, lines and ellipses using various algorithms.

**Sector**

Sector function draws and fills an elliptical pie slice.  
  
**Declaration**: void sector( int x, int y, int stangle, int endangle, int xradius, int yradius  
**Example**: Input :

x=200, y=200, x\_angle=0, e\_angle=150, x\_radius=50, y\_radius=65

**Settextstyle**

The settextstyle function is used to change the way in which text appears, using it we can modify the size of text, change direction of text and change the font of text.  
  
**Declaration**: void settextstyle( int font, int direction, int charsize);  
font argument specifies the font of text, Direction can be HORIZ\_DIR (Left to right) or VERT\_DIR (Bottom to top).

**Delay**

Delay function is used to suspend execution of a program for particular time.

**Declaration**: void delay(unsigned int);

Here unsigned int is the number of milliseconds(remember 1 second = 1000 milliseconds). To use delay function in your program you should include the “dos.h” header file which is not a part of standard C library.

**Example**: void delay(100);